# CLAIMS

1. A computer program product, tangibly embodied in an information carrier, the computer program product being operable to cause data processing apparatus to perform operations comprising:

receiving run-time code for an application, the run-time code being generated from a converted design-time representation of the application, wherein:

the converted design-time representation of the application is generated from an original design-time representation of the application developed for use in a first run-time environment for executing applications having been developed in a first design-time environment, the first design-time environment using a first programming model comprising one or more first model elements including screens and processing logic for each screen, the original design-time representation including one or more application screens and original processing logic for each application screen, the original processing logic including a call to a run-time module in the first run-time environment; and

the converted design-time representation of the application is for use in a second run-time environment for executing applications having been developed in a second design-time environment, the second design-time environment using a second programming model comprising one or more second model elements including models, views, and controllers, the converted design-time representation including one or more application views based on the one or more application screens, and converted processing logic based on the original processing logic, the converted processing logic being capable of being executed in the second run-time environment; and

executing the run-time code in the second run-time environment using an adapter operable to interface with the run-time module in the first run-time environment.

2. The computer program product of claim 1, wherein the first programming model is the SAP Dynpro programming model and the second programming model is the SAP Web Dynpro programming model.

3. The computer program product of claim 1, wherein executing the run-time code comprises using the adapter to perform a function not performed by the original processing logic.

4. The computer program product of claim 3 wherein the function comprises input validation.

5. The computer program product of claim 3 wherein the function comprises input formatting.

6. The computer program product of claim 1, wherein:

the original design-time representation of the application comprises original state control logic; and

the converted design-time representation of the application comprises converted state control logic based on the original state control logic, the converted state control logic capable of being executed by the adapter.

7. The computer program product of claim 1, wherein:

the original design-time representation of the application comprises one or more controls from a first set of controls;

the converted design-time representation of the application comprises one or more controls from a second set of controls, each control in the converted design-time representation of the application corresponding to a control in the original design-time representation of the application; and

executing the run-time code comprises rendering the controls in the converted design-time representation of the application.

8. A computer program product, tangibly embodied in an information carrier, the computer program product being operable to cause data processing apparatus to perform operations comprising:

receiving run-time code for an application;

determining whether the run-time code was generated from a native design-time representation of the application or from a converted design-time representation of the application, wherein:

the native design-time representation of the application is for use in a first run-time environment for executing applications having been developed in a first design-time environment, the first design-time environment using a first programming model comprising one or more first model elements including models, views, and controllers; and

the converted design-time representation of the application is generated from an original design-time representation of the application developed for use in a second run-time environment for executing applications having been developed in a second design-time environment, the second design-time environment using a second programming model comprising one or more second model elements including screens and processing logic for each screen, the original design-time representation including one or more application screens and original processing logic for each application screen, the converted design-time representation including one or more application views based on the one or more application screens, and converted processing logic based on the original processing logic, the converted processing logic capable of being executed in the second run-time environment; and

if the run-time code was generated from the native design-time representation, execute the run-time code in the first run-time environment using a set of run-time modules in the first run-time environment; and

if the run-time code was generated from the converted design-time representation, execute the run-time code in the first run-time environment using a set of run-time modules in the second run-time environment.

9. The computer program product of claim 8, wherein the first programming model is the SAP Web Dynpro programming model and the second programming model is the SAP Dynpro programming model.

10. The computer program product of claim 8 wherein executing the run-time code using the set of run-time modules in the second run-time environment comprises using an adapter in the first run-time environment to interface with the set of run-time modules in the second run-time environment.

11. The computer program product of claim 8, wherein:

executing the run-time code using the set of run-time modules in the first run-time environment comprises using a first sequence of process steps; and

executing the run-time code using the set of run-time modules in the second run-time environment comprises using a second sequence of process steps.

12. An apparatus comprising:

means for receiving run-time code for an application, the run-time code being generated from a converted design-time representation of the application, wherein:

the converted design-time representation of the application is generated from an original design-time representation of the application developed for use in a first run-time environment for executing applications having been developed in a first design-time environment, the first design-time environment using a first programming model comprising one or more first model elements including screens and processing logic for each screen, the original design-time representation including one or more application screens and original processing logic for each application screen, the original processing logic including a call to a run-time module in the first run-time environment; and

the converted design-time representation of the application is for use in a second run-time environment for executing applications having been developed in a second design-time environment, the second design-time environment using a second programming model comprising one or more second model elements including models, views, and controllers, the converted design-time representation including one or more application views based on the one or more application screens, and converted processing logic based on the original processing logic, the converted processing logic being capable of being executed in the second run-time environment; and

means for executing the run-time code in the second run-time environment using an adapter operable to interface with the run-time module in the first run-time environment.

13. The apparatus of claim 12, wherein the first programming model is the SAP Dynpro programming model and the second programming model is the SAP Web Dynpro programming model.

14. The apparatus of claim 12, wherein:

the original design-time representation of the application comprises original state control logic; and

the converted design-time representation of the application comprises converted state control logic based on the original state control logic, the converted state control logic capable of being executed by the adapter.

15. The apparatus of claim 12, wherein the means for executing the run-time code comprises means for using the adapter to perform a function not performed by the original processing logic.

16. A method comprising:

receiving run-time code for an application, the run-time code being generated from a converted design-time representation of the application, wherein:

the converted design-time representation of the application is generated from an original design-time representation of the application developed for use in a first run-time environment for executing applications having been developed in a first design-time environment, the first design-time environment using a first programming model comprising one or more first model elements including screens and processing logic for each screen, the original design-time representation including one or more application screens and original processing logic for each application screen, the original processing logic including a call to a run-time module in the first run-time environment; and

the converted design-time representation of the application is for use in a second run-time environment for executing applications having been developed in a second design-time environment, the second design-time environment using a second programming model comprising one or more second model elements including models, views, and controllers, the converted design-time representation including one or more application views based on the one or more application screens, and converted processing logic based on the original processing logic, the converted processing logic being capable of being executed in the second run-time environment; and

executing the run-time code in the second run-time environment using an adapter operable to interface with the run-time module in the first run-time environment.

17. The method of claim 16, wherein the first programming model is the SAP Dynpro programming model and the second programming model is the SAP Web Dynpro programming model.

18. The method of claim 16, wherein:

the original design-time representation of the application comprises original state control logic; and

the converted design-time representation of the application comprises converted state control logic based on the original state control logic, the converted state control logic capable of being executed by the adapter.

19. The method of claim 16, wherein executing the run-time code comprises using the adapter to perform a function not performed by the original processing logic.